Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 41 (2016) 393 – 398

www.elsevier.com/locate/procedia

48th CIRP Conference on MANUFACTURING SYSTEMS - CIRP CMS 2015

Modelling dependencies to improve the cross-domain collaboration in the engineering process of special purpose machinery

Tobias Helbig[a,b,*], Stefan Erler[b], Engelbert Westkämper[a], Johannes Hoos[b]

[a] *Graduate School of Excellence advanced Manufacturing Engineering (GSaME), Nobelstraße 12, 70569 Stuttgart, Germany*
[b] *Festo AG & Co.KG, Ruiter Straße 82, 73734 Esslingen, Germany*

* Corresponding author. Tel.: +49 711 347 50725; *E-mail address:* tobias.helbig@gsame.uni-stuttgart.de

Abstract

Due to individual customer demands the efficiency of the engineering process becomes more and more important for special purpose machine manufacturers. The engineering process is a cross-domain challenge between mechanics, electrics and software, but a lack of collaboration and information transfer between the domains leads to suboptimal efficiency of engineering. To improve the cross-domain collaboration this paper presents the Manufacturing System Dependency Model (MaSDeM). The basic idea of MaSDeM is to install a cross-domain solution model at the beginning of the engineering process in order to design, discuss and harmonise the principle solution with all participants involved in this project. This solution model is based on partial models, representing the performance specification by the customer, the physical layout and the detailed execution of the manufacturing steps. The partial models and their specific elements are strongly interconnected. The interconnections are realised with dependencies between the elements representing the influence that the elements have on each other. The creation of a network of dependencies provides advanced information on the system and enables the engineer to take profound decisions. This improves the cross-domain collaboration and thereby increases the efficiency of engineering. The benefit of modelling dependencies is related to the effort involved in creating and updating the model. The realisation of MaSDeM in a software tool is therefore presented concentrating on the efficient usability of the tool.

Peer-review under responsibility of the scientific committee of 48th CIRP Conference on MANUFACTURING SYSTEMS - CIRP CMS 2015

Keywords: Automation; Conceptual design; Design method; Engineering; Manufacturing; Engineering; Model

1. Introduction

Manufacturing in Europe is under great pressure from structural changes in the global economy [1]. The trend of individual consumer products leads to the necessity of tailored manufacturing systems [2]. For special purpose machinery this trend means an increasing effort for development and engineering of customised solutions. That is why efficient engineering becomes an increasingly important competitive factor. Engineering sums up the activities executed by engineers [3]. In the context of special purpose machinery engineering includes the intellectual work required for the conceptual design, the physical installation and the commissioning. It is a cross-domain challenge between mechanics, electrics and software [4]. Although each domain involved fulfils specific tasks based on their expertise the partial solutions of all domains must create one common solution to ensure the machine will be operable [5].

Nowadays the typical design process is a stepwise execution of the domain-specific design tasks which is stamped by the mechanical construction [6] (Fig 1). At the beginning the customer defines the single manufacturing steps of the product to be manufactured and any further requirements in the performance specification. Transforming these demands into a technical solution is the task of the special purpose machine manufacturer. The design process starts with the mechanical construction creating the geometrical structure and selecting the automation components. Afterwards the project is forwarded to the electrical engineer who plans the infrastructure of the machine: media and energy supply, wiring, communication and controllers. The third domain involved is

Peer-review under responsibility of the scientific committee of 48th CIRP Conference on MANUFACTURING SYSTEMS - CIRP CMS 2015
doi:10.1016/j.procir.2015.12.123

the software which defines and implements the logical behaviour of the machine.

The design phase is succeeded by the physical installation of the manufacturing system. Commissioning is the final stage of the engineering process when functionality is verified and errors are corrected. The commissioning ends with the certification by the customer.

However, difficulties often occur during commissioning of the machine when the solutions created by the different domains are consolidated and must operate together as one system [7]. Corrections executed at an existing machine during commissioning take a long time and are extremely expensive [8].

These difficulties are caused by a lack of cross-domain collaboration in the design stage [7]. There are two main reasons for this. Firstly, the domain engineers involved in the engineering process have different professional backgrounds, technical languages and engineering tools which is why exchanging information and discussing a common optimal solution is difficult [9].

The second reason is the stepwise execution of the domain-specific tasks during the design process. The mechanical engineer completely defines the geometrical structure and the components before handing the project over to the electrical engineer. Even though the geometry influences the work of the electrical engineer, he has no influence on the design created by the mechanical construction. The same problem applies to the interface between the electrical engineer and the software engineer [8].

For these reasons the design phase is not optimal. However, the conceptual design mainly defines the characteristics of the Manufacturing system, which is why special attention must be paid to this phase of engineering [8].

In this paper a model-based method for improving the engineering is proposed focusing particularly on the cross-domain collaboration and dependencies in the design phase.

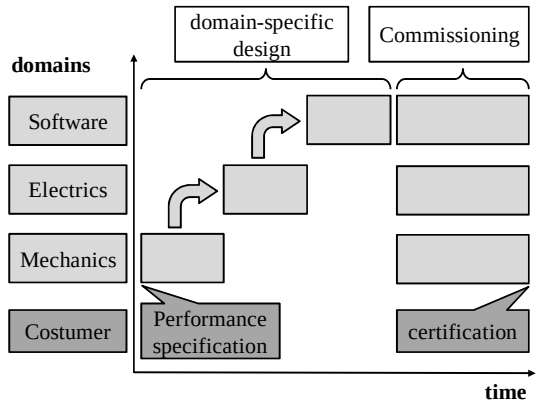

Fig 1. Typical engineering process for special purpose machinery.

2. State of the art

Faced with these problems the state of the art provides several methods aimed at improving the cross-domain collaboration. In the correspondent literature two basic ideas can be distinguished: data compatibility and simulation

2.1. Data compatibility

Exchanging data between the domain-specific engineering tools is the focus of these approaches. Providing a data interface between the tools saves time and prevents errors when transferring the data from one tool to the next [10].

The description language AutomationML [11] is an XML-based data exchange format. Each engineering tool can store and load its information in this format. This results in the complete cross-domain project being stored in AutomationML. Beyond the data of the engineering tools *Jäger et al* [12] provide a method to integrate the process description into AutomationML aiming at a seamless workflow starting with the performance specification by the customer up to commissioning.

Nevertheless exchanging data between engineering tools is not sufficient to create real cross-domain collaboration. Even more important is the exchange of ideas between the domains which means that the semantic understanding of the data must be supported. Just providing a data exchange does not automatically support the engineer to understand the ideas created in other domains.

2.2. Simulation

The basic idea of using simulation is to get reliable data on the developed system earlier in the engineering process. Instead of evaluating the operation during the commissioning, simulation methods allow the verification of the system based on the design data before the machine is physically installed. This increases the quality of information in the design phase and helps to optimise the system. Thereby errors in the design can be identified. In the design phase corrections and optimisations are less expensive to execute than during the commissioning [8].

To reduce the effort for the creation of the simulation model *Lindworsky* proposes the automatic generation of simulation models from geometrical data [4]. *Kufner* provides the generation of a simulation model with special focus on hardware in the loop simulation of the system [13].

Simulation helps to acquire feedback on the quality of the design in a fast and efficient way. Nevertheless, the simulation models can only be generated after the design of the system. This means that simulation can help finding design errors but it does not solve the problem of insufficient collaboration between the domains as it does not break up the stepwise execution of the domain-specific tasks in the design phase.

2.3. Scientific deficit

There are several approaches that aim to improve the cross-domain collaboration during the engineering process of special purpose machinery. Simulation tools mainly support the acquisition of information on the system in the early phase. The approach of exchanging data between the domain-specific tools supports the software data exchange but does not help to improve the communication and the collaboration between the domains. Both basic ideas treat the solution after the domain-specific design, either by verifying the quality of the system or by transferring the data to the next domain. Nevertheless none of the methods supports the engineer *during* the design process to design a high quality system initially. This means that the methods are not able to break up the stepwise execution of the domain-specific task leading to the problems during commissioning. Consequently they don't help to design a cross-domain optimum of the solution as they don't support the collaboration between the domains in the design stage.

Facing these deficits this paper presents a model-based engineering approach to improve the cross-domain collaboration during the design phase: the Manufacturing System Dependency Model (MaSDeM).

After introducing the structure and basic ideas of MaSDeM special attention is paid to modelling dependencies and their realisation in a software tool.

3. Manufacturing System Dependency Model (MaSDeM)

MaSDeM is a model-based method for improving the cross-domain collaboration in special purpose machinery.

The basic idea of MaSDeM is to install a cross-domain solution model (Fig 2). It serves as a common platform for all participants to create, discuss and harmonise the principle solution. All domains involved work together to develop the principle solution how the manufacturing process defined by the customer can be realised in the special purpose machine. Working on a common solution model supports the collaboration between the domains and helps to find optimal cross-domain solutions.

The development of the principle solution in the cross-domain solution model must be the initial point of the engineering process.

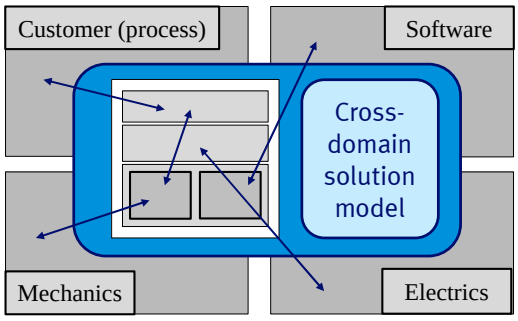

Fig 2. Basic idea of MaSDeM.

Based on the optimised principle solution represented in the solution model the further engineering process succeeds. The domain-specific detailing can be parallelised with regard to the common understanding of the system in the solution model. Each domain designs the necessary details using their expertise in the domain and the necessary specific tools.

With this method a lot of effort is invested in the early stage of engineering to create an optimal design solution. In this stage changes to the concept or the design can be executed faster, easier and cheaper than in the later phases of the engineering process [8].

The solution model must bring together the domain-specific views to the system and serve as basic for the further detailing which is why it contains a lot of dependencies, within the model and to the domain specific parts that need to be analysed and structured.

3.1. Structure of the solution model.

As shown in Fig 2 the solution model serves as the common platform for representing the principle solution. All domains participate in the definition and optimisation of this solution model.

The inner structure of the solution model distinguishes three partial models which are strongly interconnected (Fig 3): process model, station model and detail model.

The process model represents the performance specification of the customer. In this partial model the customer provides information on the product to be manufactured, the necessary manufacturing steps, the succession of these steps and functional and non-functional requirements. The elements of the partial model are the manufacturing steps which are arranged according to the procedurally possible process succession and interconnected with the requirements the specific step has to fulfil. The steps are presented in a result-oriented way meaning that only the result of the step is defined but not the way in which it is achieved.

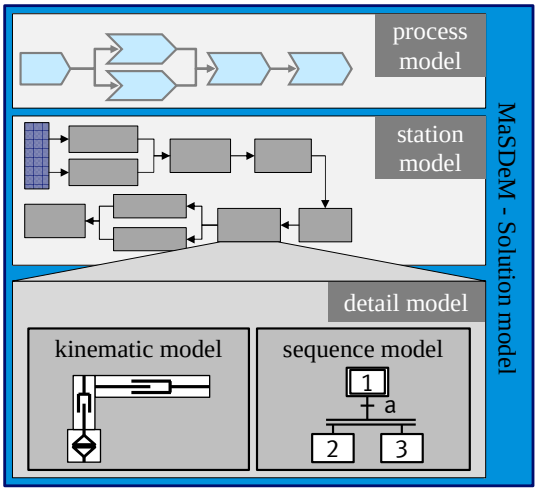

Fig 3. Structure of the MaSDeM solution model.

The station model represents the physical set-up of the manufacturing system. Each station stands for a manufacturing unit providing certain process functionality. The station model contains information on the geometrical structure and the position of the stations and the material flow between the stations. Nevertheless, a station is modelled as a black box and only characterised by its size and the functionality it offers. That is why each single station is connected to a detail model which represents how the functionality of the station can be achieved.

The detail model is the third partial model and contains two submodels: the kinematic model and the sequence model. The kinematic model represents the components of the station and the skills that the components offer. Furthermore the kinematic model contains the geometrical structure and the kinematic interconnection of the components. The sequence model defines the sequences in which the skills are executed in order to perform the defined manufacturing process of the station.

Summarising the structure of the MaSDeM cross-domain solution model, it consists of three partial models which are strongly interconnected, making the modelling of dependencies between elements necessary.

3.2. Dependencies within the solution model

Dependencies are connections between two elements representing the influence that the elements have on each other. Each dependency contains features specifying the kind of influence one element has on the other one. There are different kinds of dependencies that can be distinguished in the solution model.

The first kind of dependency is a connection within one partial model. Dependencies between objects of one partial model define the inner structure of this partial model. This can comprise the hierarchical and structural arrangement of objects of the same type. But it can also contain references representing additional information to this object. An example of modelling dependencies within the process model can be seen in Fig 4. The succession of the process steps is modelled by references showing which step must be finished before the next one can be executed. Furthermore, functional requirements (FR) can be attached to process steps, meaning that the step must meet the conditions defined in the requirement. For example the process step *turn piece* can be specified with the functional requirement *handle with care*.

References between two partial models are the second kind of dependency. Dependencies between different partial models symbolise the connection beyond the limits of the different views that the single partial models stand for. Using the example of Fig 4 dependencies between the process model and the station model appear. Connecting a process step to a station means that the station executes the referenced process step. In this example the process step *turn piece* was assigned to the *handling station* where it will be executed.

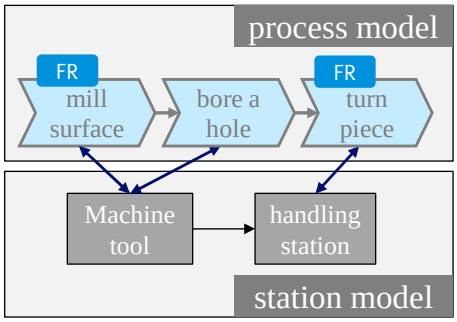

Fig 4 Dependencies between the partial models

The third kind of dependency is an interlinked dependency. The meaning of an interlinked dependency can be explained with regard to the example in Fig 5. This figure represents the detail model of the *handling station* defined in Fig 4. The Process step *turn piece* and its functional requirement *handle with care* were related to the *handling station* and must now be fulfilled in the detail model of the *handling station*.

In the detail model the components of the solution are represented: two grippers, one linear axis and one combination of linear and rotatory axis. The functional requirement *handle with care* can now directly be assigned to concrete components, here the grippers, as they are in contact with the workpiece. Thereby an interlinked dependency was created starting on the process level, but finally reaching a certain component within the detail model.

Although the interlinked objects are in different partial models a dependency can be set linking the component to the origin of the relevant information.

Analysing dependencies provides a considerable increase in the quality of information on the system and its elements. It is not limited to the direct environment but also includes the interlinked dependencies that can be analysed. Being able to take into account the influences of one element on the system leads to more profound design decisions that can be taken.

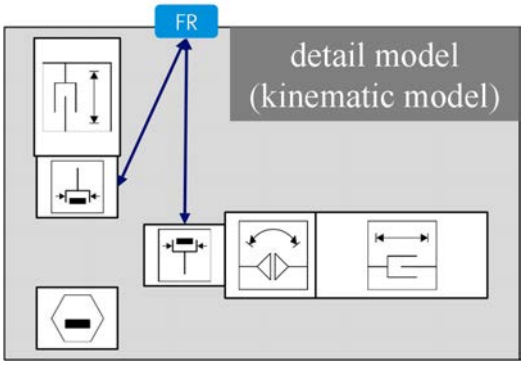

Fig 5. Detail model of the handling station.

3.3. Using dependency modelling in the domain-specific detailing

Modelling dependencies in the MaSDeM concept is not limited to the solution model. The design, discussion and harmonisation of the cross-domain solution model is succeeded by the domain-specific detailing. Therefore the information of the solution model must be transferred to the domain-specific tools (Fig 6).

In this phase the different domains work out the details of the solution using their specific expertise and their specialised engineering tools. The elements defined in the solution model appear in the domain-specific view of the system as well. This means that the structure and the dependencies that were defined in the solution model can be transferred to the domain-specific view of the specialised engineering tools.

Regarding the example of the handling station in Fig 5 the development of the principal solution was done in the solution model. In the domain-specific engineering tools the advanced information derived from the principle solution and the modelled dependencies can be used to take profound decisions on the precise features of each element. This is the basis for the selection of a suitable gripper and it can also help to apply consistent modifications to the system.

In case of changes to the requirements by the costumer the interlinked dependency directly relates the modification to the gripper and allows a verification if the selected gripper is still suitable for the new requirements.

Furthermore an optimisation within one domain can easily be discussed with the other domains based on the solution model. If the mechanical construction likes to change from a tactile gripper to a vacuum gripper, the consequences of this adaptation for media supply and sequences can easily be discussed with the other domains using the common representation in the solution model.

All results of the domain specific engineering are mirrored to the solution model. Using one common data base guarantees a seamless propagation of the information. All engineering tools read from the same data and store their design decisions in this data base which guarantees the consistency of information.

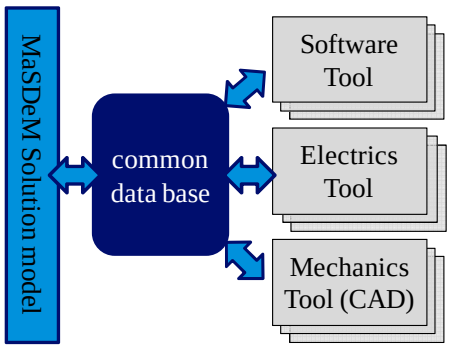

Fig 6. Data exchange between MaSDeM solution model and the domain-specific engineering tools.

3.4. Discussion of dependency modelling.

Modelling dependencies provides various benefits during the engineering process. The engineer can reach a higher information level when considering the influences on and from other elements. Using interlinked dependencies, even more information can be obtained that goes beyond the immediate environment. Based on the higher information level well-educated design decisions can be taken improving the quality of the design.

In case of errors being identified the system must be modified to solve the problem. Following the dependencies explicitly shows which other elements are affected by the correction. This helps to coordinate and discuss the impact of a modification with the colleagues involved finding a common mechatronic optimal solution.

The common data base and the data transfer from the solution model to the domain-specific tools guarantees the consistency of the information between the domains breaking up the stepwise and separated design of the domains.

Summarising it can be stated that the MaSDeM method and modelling dependencies support the cross-domain collaboration during the design process.

4. Realisation of MaSDeM in a software tool

Any method which is based on a model causes a certain effort when creating and updating the model. To amortise this effort the benefits of modelling dependencies must be higher than the effort evolved in the creation of the model [14]. To maximise the benefits of the method the modelling effort must be minimised.

The effort for creating the model is related to the quality and usability of the software tool that is used for modelling. That is why the realisation of the software tool is essential for the success of the concept.

The tool must be able to represent all elements of the model which were defined above. An even more important requirement for the tool is usability. This comprises the fast creation of models but as well the fast and easy capture of information from the model by the tool.

The tool is implemented in C#, based on a .Net-Framework. The software is realised with an object oriented pattern following the structure of the model. The model data base is separated from the user interface by an architecture consisting of three levels: Model, Viewmodel and View.

As defined in the requirements, the usability of the model is one of the key issues for this tool. The Graphical User Interface (GUI) was therefore created with the same structure across all elements and partial models.

The challenge of the arrangements is to provide information on the element and to make working on the model easy. Therefore the GUI was divided into five parts. Part 1 is the main window where the model is displayed. Furthermore there are defined fields for the creation of model elements (part 2) and for the definition of dependencies (part 3). For an overview of the model, the features and references of the chosen element

are displayed in part 4. Finally, the tree view in part 5 gives an overview of the system and the structural and hierarchical position of the element.

5. Summary

This paper presents the Manufacturing System Dependency Model (MaSDeM). The basic idea of the MaSDeM-approach is to install a cross-domain solution model in the early design stage of the engineering process. To represent the principle solution the solution model is divided into three partial models which are strongly interconnected. By modelling dependencies the engineer can reach a higher information level being able to consider the influences on and from other elements.

This supports the cross-domain collaboration in the engineering process and thereby contributes to reduce the expensive errors occurring during commissioning.

References

[1] Jovane, F., Westkämper, E., Williams, D.J., 2009. The ManuFuture road: towards competitive and sustainable high-adding-value manufacturing. Springer.
[2] Piller, F.T., 2007. Mass Customization, in: Albers, S., Herrmann, A. (Eds.), Handbuch Produktmanagement. Gabler Verlag, Wiesbaden, pp. 941-968.
[3] Westkämper, E., 2012. Engineering Apps: Eine Plattform für das Engineering in der Produktionstechnik. wt Werkstatttechnik online 102 (10), 718–722.
[4] Alexander Lindworsky, 2011. Teilautomatische Generierung von Simulationsmodellen für den entwicklungsbegleitenden Steuerungstest. Herbert Utz Verlag, München.
[5] VDI 2206, 2004. Entwicklungsmethodik für mechatronische Systeme. Beuth Verlag GmbH, Berlin 03.100.40. Accessed 8 January 2015, 118 pp.
[6] Frank, G., Westkämper, E., Schlögl, W., Lenord, M., 2013. System Design of PLC-Controlled Specialized Production Machines, in: , Smart Product Engineering. Springer, pp. 613–622.
[7] Wagner, U., 2010. Standardisierung der Projektabwicklung im kundenspezifischen Maschinen-und Anlagenbau, Chemnitz, -IX, 161, A30 S.
[8] Ehrlenspiel, K., Kiewert, A., Lindemann, U., 2007. Cost-efficient design. Springer Science & Business Media.
[9] Rauscher, M., Göhner, P., 2013. Konsistenzprüfung im frühen mechatronischen Entwurf. at - Automatisierungstechnik 61 (2), 109–113.
[10] Albrecht, H., Meyer, D., 2002. XML in der Automatisierungstechnik – Babylon des Informationsaustausches?: XML in Industrial Automation - The Angle-bracketed Writing on the Wall? at - Automatisierungstechnik 50 (2), 87–96.
[11] Faltinski, S., Niggemann, O., Moriz, N., Schetinin, N., 2012. AutomationML als Grundlage für einen durchgängigen Modellierungs-, Simulations- und Integrationsprozess in der Anlagenplanung, in: , Automation, Branchentreff der Mess- und Automatisierungstechnik, vol. 2171. VDI-Verlag, Düsseldorf, pp. 371–374.
[12] Jäger, T., Christiansen, L., Strube, M., Fay, A., 2013. Durchgängiges Engineering von der Anforderungserhebung bis zur Anlagenstrukturbeschreibung. at - Automatisierungstechnik 61 (2), 92–101.
[13] Kufner, A., 2012. Automatisierte Erstellung von Maschinenmodellen für die Hardware-in-the-Loop-Simulation von Montagemaschinen. Jost-Jetter, Heimsheim.
[14] Becker, J., 2012. Grundsätze ordnungsmäßiger Modellierung. Springer, Berlin, Heidelberg.